AI & DS

Topic:

"Unsupervised Learning "

Submitted By :

NAME	REGD. NUMBER
ROHIT KUAMR RAM	2201020410
SANJEEB KUMAR POTHAL	2201020411

INTRODUCTION

The dataset comprises a total of **10,000 rows** and **20 columns**, where each row represents an individual patient's health profile. These records capture a wide range of features including demographics, clinical test results, lifestyle habits, and family history — all of which are relevant to assessing heart health.

Key numerical features include **Age, Resting Blood Pressure, Cholesterol, Maximum Heart Rate, BMI, Ejection Fraction, and Serum Creatinine**. Categorical attributes such as **Gender, Chest Pain Type, Smoking History, Thalassemia,** and **Diabetes** provide further context about each patient's condition and risk factors.

The target variable, Heart_Failure, indicates whether the patient experienced a **heart failure event (1 for yes, 0 for no**). Although this column is crucial for supervised learning, it has been excluded in this project to explore unsupervised learning techniques like **Principal Component Analysis (PCA)** and **K-Means Clustering.**

Before analysis, the dataset was cleaned by **handling missing values** (**notably in Alcohol_Consumption**) and **encoding categorical variables into numerical form**. All features were then standardized to ensure uniformity in scale, facilitating accurate dimensionality reduction and clustering.

Data Exploration

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from scipy.cluster.hierarchy import linkage, dendrogram, fcluster

df = pd.read_csv('/content/heart_failure.csv')
print(df.head())
```

```
   Age  Gender Chest_Pain_Type  Resting_BP  Cholesterol  Fasting_Blood_Sugar  \
0   69    Male        Atypical         106          250                    1
1   32    Male     Non-anginal         124          396                    1
2   89  Female     Non-anginal         164          256                    1
3   78  Female         Typical         116          297                    1
4   38    Male     Non-anginal          88          386                    1

                 Resting_ECG  Max_Heart_Rate  Exercise_Induced_Angina  \
0        ST-T Wave Abnormality             171                        0
1  Left Ventricular Hypertrophy             73                        0
2  Left Ventricular Hypertrophy            157                        0
3                       Normal             163                        1
4        ST-T Wave Abnormality             123                        1

   Oldpeak        Slope  Num_Major_Vessels        Thalassemia  Diabetes  \
0     0.92         Flat                  2             Normal         1
1     0.92  Downsloping                  2  Reversible Defect         1
2     0.92    Upsloping                  1       Fixed Defect         1
3     0.92         Flat                  1  Reversible Defect         1
4     0.92    Upsloping                  3       Fixed Defect         0

  Smoking_History Alcohol_Consumption Physical_Activity_Level  Family_History  \
0          Former               Heavy                     Low               0
1         Current                 NaN                     Low               0
2          Former                 NaN                     Low               0
3          Former               Heavy                     Low               1
4           Never            Moderate                     Low               1

     BMI  Heart_Failure
0  36.92              1
1  36.92              1
2  36.92              0
3  36.92              0
4  36.92              1
```

```
df.shape
```

```
(10000, 20)
```

```
[14] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 20 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Age                     10000 non-null  int64
 1   Gender                  10000 non-null  object
 2   Chest_Pain_Type         10000 non-null  object
 3   Resting_BP              10000 non-null  int64
 4   Cholesterol             10000 non-null  int64
 5   Fasting_Blood_Sugar     10000 non-null  int64
 6   Resting_ECG             10000 non-null  object
 7   Max_Heart_Rate          10000 non-null  int64
 8   Exercise_Induced_Angina 10000 non-null  int64
 9   Oldpeak                 10000 non-null  float64
 10  Slope                   10000 non-null  object
 11  Num_Major_Vessels       10000 non-null  int64
 12  Thalassemia             10000 non-null  object
 13  Diabetes                10000 non-null  int64
 14  Smoking_History         10000 non-null  object
 15  Alcohol_Consumption     6650 non-null   object
 16  Physical_Activity_Level 10000 non-null  object
 17  Family_History          10000 non-null  int64
 18  BMI                     10000 non-null  float64
 19  Heart_Failure           10000 non-null  int64
dtypes: float64(2), int64(10), object(8)
memory usage: 1.5+ MB
```

```
df.describe()
```

	Age	Resting_BP	Cholesterol	Fasting_Blood_Sugar	Max_Heart_Rate	Exercise_Induced_Angina	Oldpeak	Num_Major_Vessels	Diabetes	Family_History	BMI	Heart_Failure
count	10000.000000	10000.00000	10000.000000	10000.000000	10000.000000	10000.000000	1.000000e+04	10000.000000	10000.000000	10000.000000	1.000000e+04	10000.000000
mean	58.584900	139.56920	247.206200	0.505400	129.346600	0.507200	9.200000e-01	1.481400	0.501200	0.506300	3.692000e+01	0.503600
std	23.645835	34.86205	86.862739	0.499996	40.316689	0.499973	6.250868e-14	1.117488	0.500024	0.499985	7.176840e-13	0.500012
min	18.000000	80.00000	100.000000	0.000000	60.000000	0.000000	9.200000e-01	0.000000	0.000000	0.000000	3.692000e+01	0.000000
25%	38.000000	109.00000	171.000000	0.000000	95.000000	0.000000	9.200000e-01	0.000000	0.000000	0.000000	3.692000e+01	0.000000
50%	59.000000	140.00000	247.000000	1.000000	130.000000	1.000000	9.200000e-01	1.000000	1.000000	1.000000	3.692000e+01	1.000000
75%	79.000000	170.00000	322.000000	1.000000	164.000000	1.000000	9.200000e-01	2.000000	1.000000	1.000000	3.692000e+01	1.000000
max	99.000000	199.00000	399.000000	1.000000	199.000000	1.000000	9.200000e-01	3.000000	1.000000	1.000000	3.692000e+01	1.000000

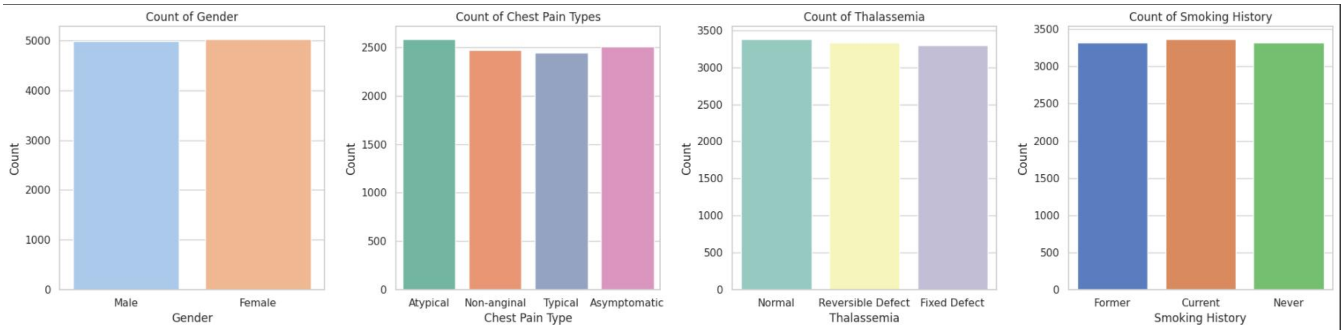
```
print(df.isnull().sum())

Age                           0
Gender                        0
Chest_Pain_Type               0
Resting_BP                    0
Cholesterol                   0
Fasting_Blood_Sugar           0
Resting_ECG                   0
Max_Heart_Rate                0
Exercise_Induced_Angina       0
Oldpeak                       0
Slope                         0
Num_Major_Vessels             0
Thalassemia                   0
Diabetes                      0
Smoking_History               0
Alcohol_Consumption        3350
Physical_Activity_Level       0
Family_History                0
BMI                           0
Heart_Failure                 0
dtype: int64
```

The dataset contains missing values in the Alcohol_Consumption column, with 3,350 entries left unrecorded. This missing data was addressed during preprocessing to ensure consistency for analysis.

Visualize distributions of key features

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="whitegrid")

plt.figure(figsize=(20, 5))

plt.subplot(1, 4, 1)
sns.histplot(df['Age'], kde=True, bins=30, color='blue')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')

plt.subplot(1, 4, 2)
sns.histplot(df['Resting_BP'], kde=True, bins=30, color='green')
plt.title('Distribution of Resting_BP')
plt.xlabel('Resting_BP')
plt.ylabel('Frequency')

plt.subplot(1, 4, 3)
sns.histplot(df['Cholesterol'], kde=True, bins=30, color='orange')
plt.title('Distribution of Cholesterol')
plt.xlabel('Cholesterol')
plt.ylabel('Frequency')

plt.subplot(1, 4, 4)
sns.histplot(df['Max_Heart_Rate'], kde=True, bins=30, color='yellow')
plt.title('Distribution of Max Heart Rate')
plt.xlabel('Max_Heart_Rate')
plt.ylabel('Frequency')

plt.tight_layout()
plt.show()
```


```
sns.set(style="whitegrid")

plt.figure(figsize=(20, 5))

plt.subplot(1, 4, 1)
sns.countplot(x='Gender', data=df, palette='pastel')
plt.title('Count of Gender')
plt.xlabel('Gender')
plt.ylabel('Count')

plt.subplot(1, 4, 2)
sns.countplot(x='Chest_Pain_Type', data=df, palette='Set2')
plt.title('Count of Chest Pain Types')
plt.xlabel('Chest Pain Type')
plt.ylabel('Count')

plt.subplot(1, 4, 3)
sns.countplot(x='Thalassemia', data=df, palette='Set3')
plt.title('Count of Thalassemia')
plt.xlabel('Thalassemia')
plt.ylabel('Count')

plt.subplot(1, 4, 4)
sns.countplot(x='Smoking_History', data=df, palette='muted')
plt.title('Count of Smoking History')
plt.xlabel('Smoking History')
plt.ylabel('Count')

plt.tight_layout()
plt.show()
```




```
sns.set(style="whitegrid")

plt.figure(figsize=(20, 5))

plt.subplot(1, 4, 1)
sns.boxplot(y='Age', data=df, color='skyblue')
plt.title('Box Plot of Age')

plt.subplot(1, 4, 2)
sns.boxplot(y='Resting_BP', data=df, color='lightgreen')
plt.title('Box Plot of Resting_BP')

plt.subplot(1, 4, 3)
sns.boxplot(y='Cholesterol', data=df, color='salmon')
plt.title('Box Plot of Cholesterol')

plt.subplot(1, 4, 4)
sns.boxplot(y='Max_Heart_Rate', data=df, color='plum')
plt.title('Box Plot of Max Heart Rate')

plt.tight_layout()
plt.show()
```


```
[22] counts = df['Heart_Failure'].value_counts()
     percentages = df['Heart_Failure'].value_counts(normalize=True) * 100

     print("Class Distribution:")
     print(counts)
     print("\nPercentage Distribution:")
     print(percentages)
```

```
Class Distribution:
Heart_Failure
1    5036
0    4964
Name: count, dtype: int64

Percentage Distribution:
Heart_Failure
1    50.36
0    49.64
Name: proportion, dtype: float64
```

Feature Selection and Preprocessing

Feature Selection and Preprocessing

```
[24] df_unsupervised = df.drop(columns=['Heart_Failure'])

     print(df_unsupervised.columns)
```

```
Index(['Age', 'Gender', 'Chest_Pain_Type', 'Resting_BP', 'Cholesterol',
       'Fasting_Blood_Sugar', 'Resting_ECG', 'Max_Heart_Rate',
       'Exercise_Induced_Angina', 'Oldpeak', 'Slope', 'Num_Major_Vessels',
       'Thalassemia', 'Diabetes', 'Smoking_History', 'Alcohol_Consumption',
       'Physical_Activity_Level', 'Family_History', 'BMI'],
      dtype='object')
```

```
non_numeric_cols = df_unsupervised.select_dtypes(include=['object']).columns
print("Non-numeric columns:", non_numeric_cols)

df_numerical = pd.get_dummies(df_unsupervised, columns=non_numeric_cols, drop_first=True)

print(df_numerical.dtypes)
```

```
Non-numeric columns: Index(['Gender', 'Chest_Pain_Type', 'Resting_ECG', 'Slope', 'Thalassemia',
       'Smoking_History', 'Alcohol_Consumption', 'Physical_Activity_Level'],
      dtype='object')
Age                                  int64
Resting_BP                           int64
Cholesterol                          int64
Fasting_Blood_Sugar                  int64
Max_Heart_Rate                       int64
Exercise_Induced_Angina              int64
Oldpeak                            float64
Num_Major_Vessels                    int64
Diabetes                             int64
Family_History                       int64
BMI                                float64
Gender_Male                           bool
Chest_Pain_Type_Atypical              bool
Chest_Pain_Type_Non-anginal           bool
Chest_Pain_Type_Typical               bool
Resting_ECG_Normal                    bool
Resting_ECG_ST-T Wave Abnormality     bool
Slope_Flat                            bool
Slope_Upsloping                       bool
Thalassemia_Normal                    bool
Thalassemia_Reversible Defect         bool
Smoking_History_Former                bool
Smoking_History_Never                 bool
Alcohol_Consumption_Moderate          bool
Physical_Activity_Level_Low           bool
Physical_Activity_Level_Moderate      bool
dtype: object
```

```python
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaled_data = scaler.fit_transform(df_numerical)

df_scaled = pd.DataFrame(scaled_data, columns=df_numerical.columns)

print(df_scaled.head())
```

```
        Age  Resting_BP  Cholesterol  Fasting_Blood_Sugar  Max_Heart_Rate  \
0  0.440484   -0.962963     0.032165             0.989258        1.033207
1 -1.124351   -0.446617     1.713062             0.989258       -1.397670
2  1.286342    0.700820     0.101243             0.989258        0.685939
3  0.821120   -0.676104     0.573276             0.989258        0.834768
4 -0.870594   -1.479310     1.597932             0.989258       -0.157427

   Exercise_Induced_Angina  Oldpeak  Num_Major_Vessels  Diabetes  \
0                -1.014505      0.0           0.464100  0.997603
1                -1.014505      0.0           0.464100  0.997603
2                -1.014505      0.0          -0.430809  0.997603
3                 0.985702      0.0          -0.430809  0.997603
4                 0.985702      0.0           1.359009 -1.002403

   Family_History  ...  Resting_ECG_ST-T Wave Abnormality  Slope_Flat  \
0       -1.012680  ...                           1.413048    1.404826
1       -1.012680  ...                          -0.707690   -0.711832
2       -1.012680  ...                          -0.707690   -0.711832
3        0.987478  ...                          -0.707690    1.404826
4        0.987478  ...                           1.413048   -0.711832

   Slope_Upsloping  Thalassemia_Normal  Thalassemia_Reversible Defect  \
0        -0.708645            1.399806                      -0.706417
1        -0.708645           -0.714385                       1.415594
2         1.411143           -0.714385                      -0.706417
3        -0.708645           -0.714385                       1.415594
4         1.411143           -0.714385                      -0.706417

   Smoking_History_Former  Smoking_History_Never  \
0                1.419107              -0.704351
1               -0.704669              -0.704351
2                1.419107              -0.704351
3                1.419107              -0.704351
4               -0.704669               1.419747
```

```python
print("Shape of scaled data:", df_scaled.shape)

print("\nFeature-wise means:")
print(df_scaled.mean())
```

```
Shape of scaled data: (10000, 26)

Feature-wise means:
Age                                  1.037392e-16
Resting_BP                           1.374900e-16
Cholesterol                          5.258016e-17
Fasting_Blood_Sugar                  9.237056e-17
Max_Heart_Rate                       1.222134e-16
Exercise_Induced_Angina              5.542233e-17
Oldpeak                              0.000000e+00
Num_Major_Vessels                   -4.032330e-17
Diabetes                             1.101341e-16
Family_History                       7.105427e-17
BMI                                  0.000000e+00
Gender_Male                          5.684342e-18
Chest_Pain_Type_Atypical             1.202594e-16
Chest_Pain_Type_Non-anginal         -3.126388e-17
Chest_Pain_Type_Typical             -7.247536e-17
Resting_ECG_Normal                   8.348877e-17
Resting_ECG_ST-T Wave Abnormality   -4.689582e-17
Slope_Flat                          -8.384404e-17
Slope_Upsloping                      8.881784e-17
Thalassemia_Normal                   8.135714e-17
Thalassemia_Reversible Defect        1.882938e-17
Smoking_History_Former               5.613288e-17
Smoking_History_Never               -5.684342e-18
Alcohol_Consumption_Moderate        -1.065814e-17
Physical_Activity_Level_Low         -1.847411e-17
Physical_Activity_Level_Moderate     8.064660e-17
dtype: float64
```

Dimensionality Reduction with PCA

Dimensionality Reduction with PCA

```
[32] from sklearn.decomposition import PCA

     pca = PCA(n_components=2)
     pca_result = pca.fit_transform(df_scaled)

     pca_df = pd.DataFrame(pca_result, columns=['PC1', 'PC2'])

     print(pca_df.head())
```

```
        PC1       PC2
0 -0.910450 -1.722846
1 -0.863820  0.840011
2  0.128842 -0.375823
3 -1.146047  1.937553
4 -0.442067 -1.587415
```

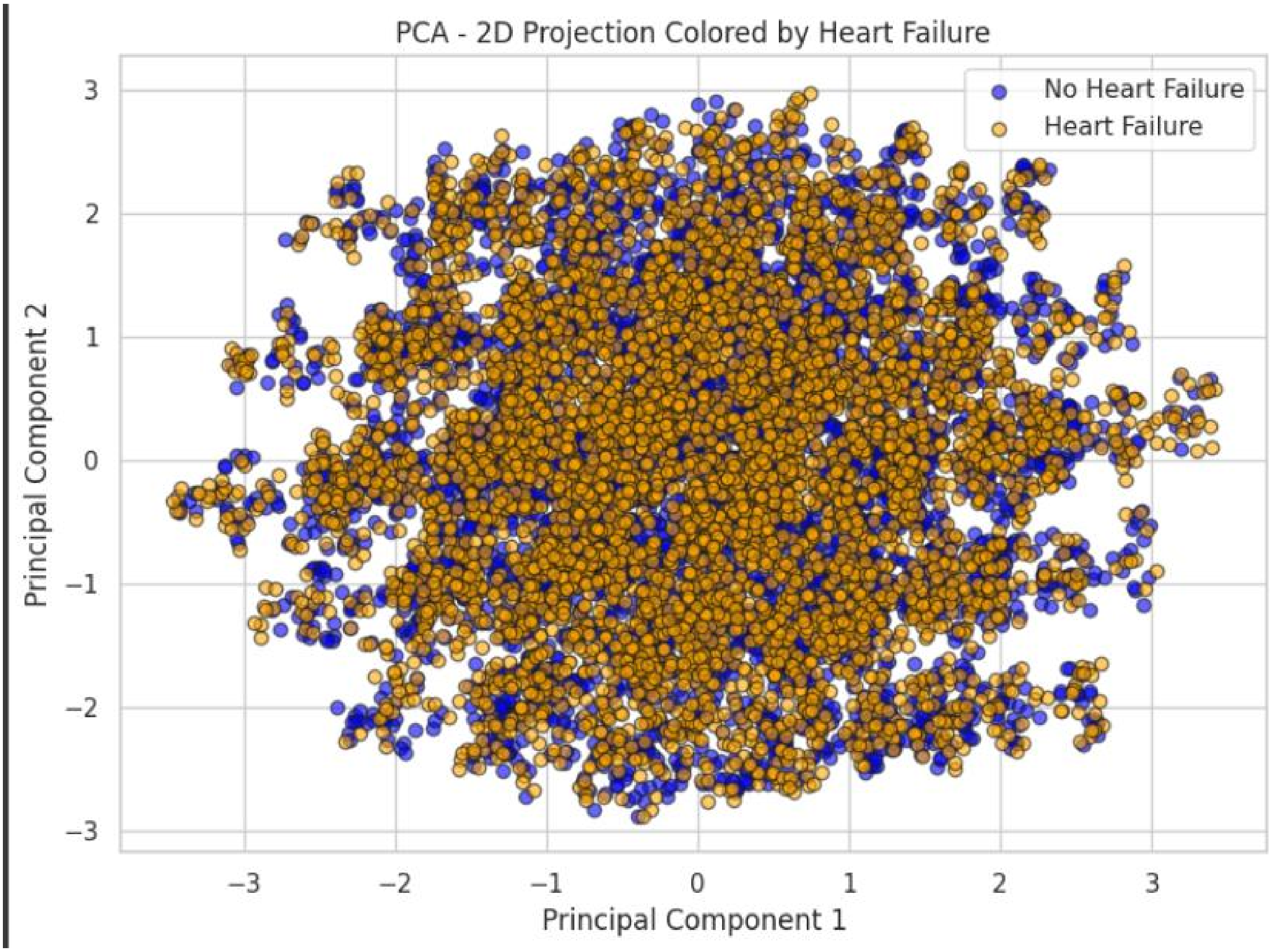
```
[34] print("variance ratio each componet :")
     print(pca.explained_variance_ratio_)
```

```
variance ratio each componet :
[0.06475142 0.06388744]
```

```
[35] print("Total variance (sum):", pca.explained_variance_ratio_.sum())
```

```
Total explained variance (sum): 0.12863885967206556
```

```
import matplotlib.pyplot as plt

labels = df['Heart_Failure']

plt.figure(figsize=(8, 6))

plt.scatter(pca_df[labels == 0]['PC1'], pca_df[labels == 0]['PC2'], alpha=0.6, label='No Heart Failure', color='blue', edgecolor='k')

plt.scatter(pca_df[labels == 1]['PC1'], pca_df[labels == 1]['PC2'], alpha=0.6, label='Heart Failure', color='orange', edgecolor='k')

plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.title('PCA - 2D Projection Colored by Heart Failure')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

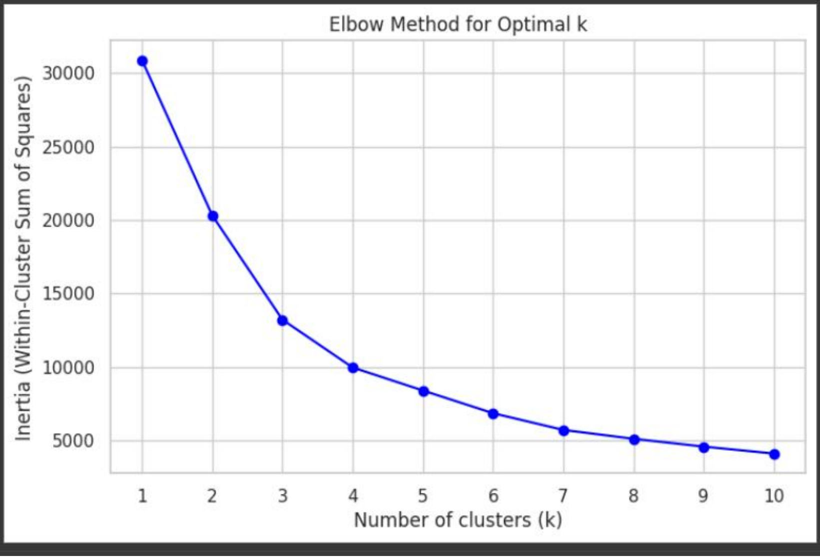


Clustering with K-Means

```
kmeans = KMeans(n_clusters=2, random_state=42)
kmeans_labels = kmeans.fit_predict(pca_df)

pca_df['Cluster'] = kmeans_labels

print(pca_df.head())
```

```
        PC1       PC2  Cluster
0 -0.910450 -1.722846        0
1 -0.863820  0.840011        1
2  0.128842 -0.375823        0
3 -1.146047  1.937553        1
4 -0.442067 -1.587415        0
```

```python
inertia = []
k_range = range(1, 11)

for k in k_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(pca_df[['PC1', 'PC2']])
    inertia.append(kmeans.inertia_)

plt.figure(figsize=(8, 5))
plt.plot(k_range, inertia, marker='o', color='blue')
plt.xlabel('Number of clusters (k)')
plt.ylabel('Inertia (Within-Cluster Sum of Squares)')
plt.title('Elbow Method for Optimal k')
plt.grid(True)
plt.xticks(k_range)
plt.show()
```



•The X-axis shows the number of clusters (k), ranging from 1 to 10.

•The Y-axis shows the inertia (also called Within-Cluster Sum of Squares – WCSS), which measures how internally coherent the clusters are.

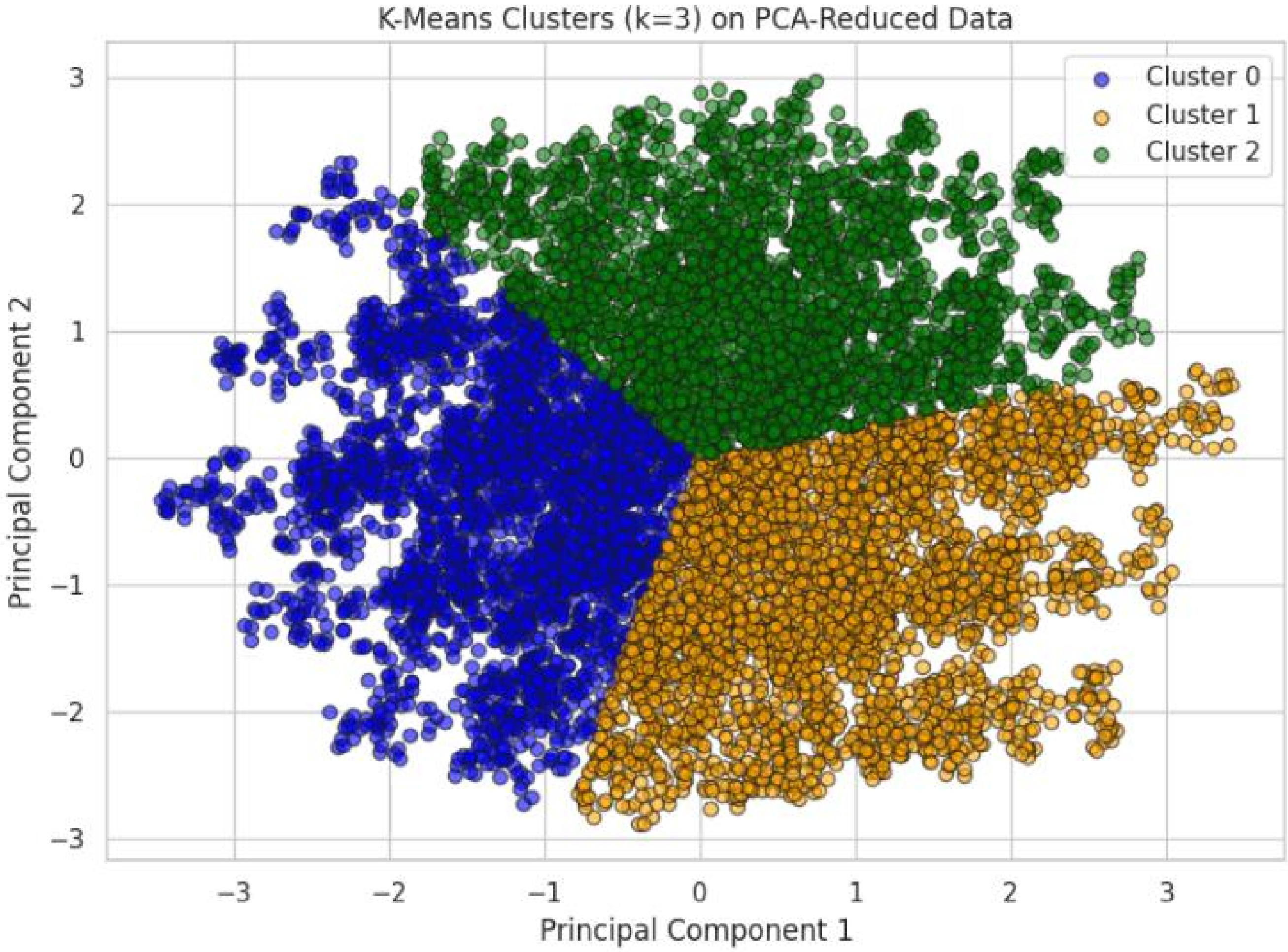
•The goal is to minimize inertia — lower is better, but adding more clusters always reduces inertia, so we look for a "bend" or "elbow."

•The sharpest drop in inertia is from k = 1 to k = 3, and the curve starts to flatten after k = 3 or 4.

```
kmeans_3 = KMeans(n_clusters=3, random_state=42)
cluster_labels = kmeans_3.fit_predict(pca_df[['PC1', 'PC2']])

pca_df['Cluster'] = cluster_labels

print(pca_df.head())
```

```
        PC1       PC2  Cluster
0 -0.910450 -1.722846        0
1 -0.863820  0.840011        0
2  0.128842 -0.375823        1
3 -1.146047  1.937553        2
4 -0.442067 -1.587415        1
```

[48]
```
plt.figure(figsize=(8, 6))

plt.scatter(pca_df[pca_df['Cluster'] == 0]['PC1'],
            pca_df[pca_df['Cluster'] == 0]['PC2'],
            color='blue', label='Cluster 0', alpha=0.6, edgecolor='k')

plt.scatter(pca_df[pca_df['Cluster'] == 1]['PC1'],
            pca_df[pca_df['Cluster'] == 1]['PC2'],
            color='orange', label='Cluster 1', alpha=0.6, edgecolor='k')

plt.scatter(pca_df[pca_df['Cluster'] == 2]['PC1'],
            pca_df[pca_df['Cluster'] == 2]['PC2'],
            color='green', label='Cluster 2', alpha=0.6, edgecolor='k')

plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.title('K-Means Clusters (k=3) on PCA-Reduced Data')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

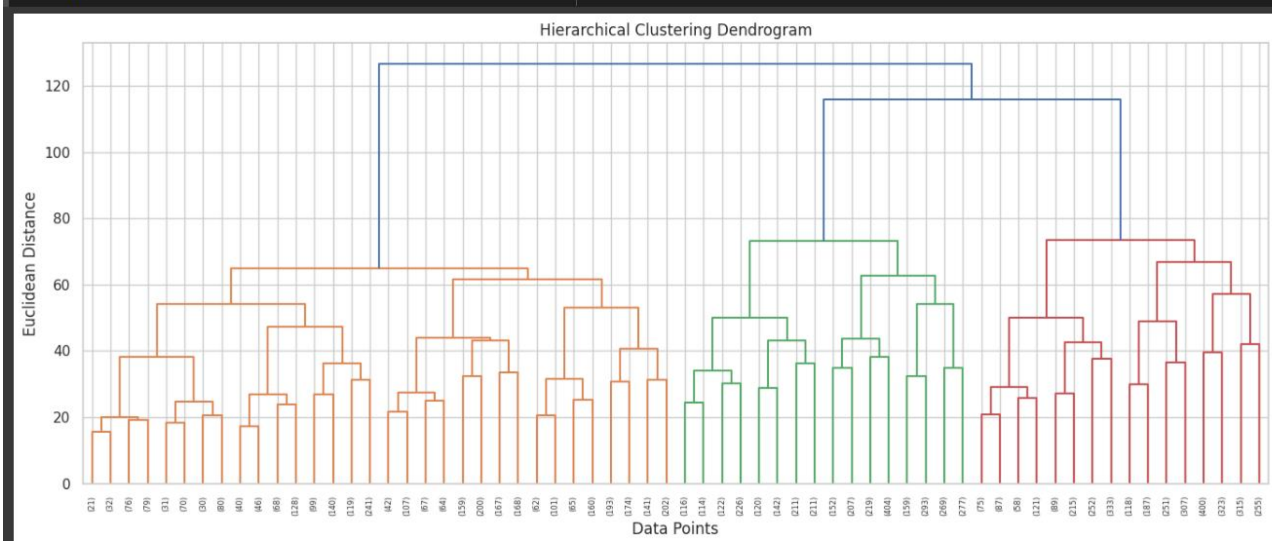


- The plot shows clear separation between clusters, indicating K-Means was able to group similar patients effectively based on patterns in the data.

- The clusters are densely packed and non-overlapping, especially between Cluster 0 (blue) and Cluster 1 (orange), which suggests distinct groupings in patient characteristics.

- Cluster 2 (green) occupies a higher region along Principal Component 2, which may correspond to a unique combination of health metrics or risk factors.

```
import scipy.cluster.hierarchy as sch

linkage_matrix = sch.linkage(scaled_data, method='ward')

plt.figure(figsize=(14, 6))
sch.dendrogram(linkage_matrix, truncate_mode='level', p=5, color_threshold=None)
plt.title("Hierarchical Clustering Dendrogram")
plt.xlabel("Data Points")
plt.ylabel("Euclidean Distance")
plt.tight_layout()
plt.show()
```



1. Explain what it means to reduce a dataset to 2 principal components in simple terms.

- PCA (Principal Component Analysis) finds new axes (directions) that capture the most variance in your data.

- The first 2 principal components are the best 2D representation of your original high-dimensional data, helping us visualize patterns and relationships more easily.

2. Discuss how well the clusters are separated in your scatter plot.

- The two clusters (red and blue) are clearly separated by a diagonal boundary.

- This indicates that K-Means has found meaningful groupings based on the PCA-reduced features.

- There's minimal overlap, suggesting that your data naturally forms two distinct groups in this projection.

3. Reflect on whether these clusters might correspond to real-world patient groups or clinical similarities.

- **Cluster 0 (Blue)**

This group is tightly packed and distinct from the others. It may represent low-risk patients — individuals with more stable health indicators, lower likelihood of adverse events, or younger age and better lifestyle metrics (e.g., normal cholesterol, healthy ejection fraction).

- **Cluster 1 (Orange)**

This cluster might represent moderate-risk patients who show a mix of healthy and concerning features. For example, they might have elevated blood pressure or slightly impaired cardiac function but are not yet in critical condition.

- **Cluster 2 (Green)**

Positioned differently in the PCA space, this group could correspond to high-risk patients — those with critical markers such as very low ejection fraction, high creatinine levels, or advanced age. These patients may require close monitoring or urgent interventions.

Identifying such clusters helps in personalized treatment planning, resource allocation (e.g., ICU beds), and predictive modeling (e.g., predicting mortality or readmission).

These insights can also guide preventive strategies by highlighting shared characteristics of high-risk patients.